Software Engineering Institute | Carnegie Mellon University

SEI SPRUCE Project

Curating Recommended Practices for Software Producibility

Mike Konrad
Craig Meyers
Tamara Marshall
Jerry Miller
Bill Pollak

January 2015

EXECUTIVE SUMMARY

This paper describes the work that the Software Engineering Institute (SEI) performed on the Systems and Software Producibility Collaboration Environment (SPRUCE) from June 2013 to June 2014. The SEI pursued an alternative but complementary approach to the original Challenge Problem–Candidate Solution strategy of SPRUCE by creating content that would engage practitioners as well as researchers. The project resulted in recommended practices on five software topics: *Agile at Scale*, *Safety-Critical Systems*, *Monitoring Software-Intensive System Acquisition Programs*, *Managing Intellectual Property in the Acquisition of Software-Intensive Systems*, and *Managing Operational Resilience*. These five practices were published as web pages, and webinars were released on two of the topics. This paper describes this expanded vision for SPRUCE; an approach to curating recommended practices on complex topics and making them available to a broad community; and the results achieved in developing the practices, disseminating them to the public, and developing a curation process. The paper ends with a suggested way forward to continue to engage the community.

HISTORICAL CONTEXT

Well-defined challenge problems related to Department of Defense (DoD) needs in software producibility should spark innovation in software producibility and drive engineering research forward. However, the failure to properly articulate such challenges and then to marshal resources to address them has contributed to continuing problems with developing large software-intensive systems for the DoD within schedule and budget [Lardieri 2009]. The objective of the SPRUCE project in its earliest phase was to provide an open, collaborative research and development environment in which to demonstrate and evaluate the ability of new techniques and technologies to assist in the affordable, predictable production of software-intensive systems [Lardieri 2009]. To meet

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

Phone: 412-268-5800
Toll-free: 1-888-201-4479

www.sei.cmu.edu

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **JAN 2015**	2. REPORT TYPE	3. DATES COVERED **00-00-2015 to 00-00-2015**

4. TITLE AND SUBTITLE
SEI SPRUCE Project: Curating Recommended Practices for Software Producibility

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Carnegie Mellon University,Software Engineering Institute,Pittsburgh,PA,15213

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a REPORT **unclassified**	b ABSTRACT **unclassified**	c THIS PAGE **unclassified**	**Same as Report (SAR)**	**18**	

this objective, the SPRUCE collaboration environment would enable stakeholders to discuss challenge problems and evaluate potential solutions in an experimentation environment containing software artifacts and computing systems that promote scientifically rigorous evaluation [CSIAC 2014].

Initially, SPRUCE aimed to facilitate the development of research products and methods for software-intensive systems and to enable universities and industry to both contribute to and leverage from the development of such technology. Phase 1 consisted of defining, developing, and documenting a concept of operations and system architecture for SPRUCE to meet the program objectives. Phase 2 was an 18-month program to build and deploy the infrastructure for the portal and populate it with an initial set Challenge Problems, Candidate Solutions, Experiments, and Communities of Interest. Phase 3 was an 18-month program to expand the content and community participation. SPRUCE began to use community moderators in focus areas to both contribute and solicit the community to contribute challenge problems. This phase achieved its goals in populating data through moderator-contributed content but did not meet targeted user registrations or community contributions.

In a quest to improve the volume of community contributions, program stakeholders decided to transfer the hosting of SPRUCE from a DoD contractor, Lockheed Martin, to a neutral institution, such as a Federally Funded Research and Development Center with a history of service to the software engineering community and an existing user base. SPRUCE Phase 4 was a 13-month program to transition the portal operations to the Cyber Security and Information Systems Information Analysis Center (CSIAC) and the content development and moderation strategy to the SEI [Lockheed Martin 2013].

CURATING RECOMMENDED PRACTICES

SEI Vision for SPRUCE

If a software developer, acquirer, or sustainer has a question about best practices related to software—for example, “What is the best way to perform software cost estimation?”—there is no single source of expertise from which to obtain high-quality advice. Technical experts are often called on to offer their expertise to stakeholders seeking advice on particular topics, but summarizing vast amounts of research and experience is difficult. Technical experts regard the importance of people, process, tool, environment, organizational, and contracting factors differently, making the resulting synthesis of advice idiosyncratic. Furthermore, all the information relevant to a particular topic does not exist in one convenient place. Instead, the questioner must find incomplete advice from multiple sources that can be difficult to reconcile and implement in a particular context. There is clearly value in the multiple technical perspectives and experiences that different experts bring to a topic, but meshing that advice into something that is coherent, applicable, and transferrable is a challenge.

The SEI’s vision for SPRUCE was for it to become a virtual meeting place for learning about and contributing to recommended practices on critical software topics. We wanted to create a *curated* collection of recommended practices. Rather than merely collecting and maintaining practices, we would engage technical experts across the broader communities of software-intensive system acquisition and development to reflect on their varied experiences and synthesize recommended practices on particular software topics.

Given the original focus for SPRUCE on exploring software problems and solutions, we would curate the practices in the context of a community that is concerned about deeper programmatic and engineering challenges in software-intensive system development. Recommended practices are a step in the community learning and maturation process. First, the community and experts identify persistent challenges; second, the community explores potential solutions to those problems; third, it develops recommended practices that implement a solution in a broad variety of contexts; finally, the practices mature into standards that typically represent the state of the practice. (Of course, this process is idealized—these steps often cannot be pursued sequentially, which was a motivation for creating SPRUCE in the first place.) The CSIAC's SPRUCE website would host the curated recommended practices in the broader context of the original SPRUCE vision.

Approach

Producing valued content was the most important goal for the SEI's contribution to SPRUCE. We chose topics based on our perceptions of what the software engineering community most needed, the maturity of knowledge about the topic, and timeliness; for example, we selected managing operational resilience as a topic shortly after the Target Corporation experienced a security breach. We planned to produce one set of recommended practices per month until we had published web pages on five software topics. Having a small collection of high-quality content for visitors to browse and engage with was a prerequisite to seeking broader transition of the work by promoting the website.

We established a process to choose a topic and identify technical experts at the SEI who had significant experience with that topic, such as through multiple client engagements or research projects, and had published extensively on that topic. We asked these experts to explain the importance of the topic, recommend up to 10 practices, identify challenges and enablers to their effective implementation, and identify sources for further reading. We then interviewed the experts, drafted the recommended practices, augmented the recommendations and resources through literature searches, and sent the draft to the experts for review and revision. We sent the revised practices to Quanterion, the CSIAC's partner, for posting to the SPRUCE website.

Once we had an initial set of recommended practices for five software topics, we began to develop approaches for a promotion campaign to announce the site, draw more traffic to it, and provide a means of interaction. We also planned to use derivative works such as webinars to promote the site, based on the observation that the number of visitors to the recommended practices temporarily increased after webinars on *Agile at Scale* and *Safety-Critical Systems*.[1]

[1] Available at http://vimeo.com/81636494 and http://vimeo.com/86733718.

Structure of the Recommended Practices

Organizations will experience challenges when attempting to address the topics covered by the recommended practices, and organizations with certain characteristics (e.g., effective training programs and mature organizational processes) will have an easier time overcoming those challenges. Therefore, in addition to the recommended practices, we included descriptions of those challenges as well as example enablers that help an organization overcome those challenges and effectively implement the recommended practices. The practices also include references to additional resources in two ways. First, throughout each web page, we link to references in the text to help amplify or validate a point. However, each web page stands on its own—these resources are not necessary to understand the recommended practices but provide more detailed information (e.g., on how to implement a particular concept). Second, we list more resources at the end of each web page, curated from those provided by the technical experts we interviewed as well as our supplemental research. This resource section is divided by subtopic, with a few resources for each subtopic.

The recommended practices were published as web pages on the CSIAC web site [SPRUCE 2013]. While Quanterion made some adjustments to the website design, for the most part, we needed to work within the website's existing style libraries and operational models for web page structure, navigation, and commenting.

The web pages had a Comments feature, which allows registered, logged-in visitors to comment on the practices. We hoped this would increase community engagement and lead to further refinement of the practices, but visitors did not use this feature. (For more detail, see the Evaluating Impact section.)

RESULTS

Recommended Practices

This project resulted in recommended practices on five software topics, published as five web pages, each of which has a similar structure based on the initial design. The appendix contains screenshots of the home page for the practices and a sample web page for one topic. Each web page contains the following information:

- Discussion of what makes the topic challenging and why it is relevant
- Recommended practices
- Steps an organization can take to more effectively implement the practices
- Resources that will help readers learn more about the topic, organized by subtopic

Agile at Scale

The recommended practices area of the SPRUCE website launched in September 2013 with the practices for *Agile at Scale*.[2] The practices were based on interviews with Ipek Ozkaya and Robert Nord. Dr. Ozkaya is a technical lead for research on architecture-aware methods and techniques for designing, analyzing, and evolving software systems iteratively and incrementally. And Dr. Nord leads research in strategies for scaling agile development by incorporating architecture practices and has co-authored two books on software architecture.

Agile practices have been used for well over a decade and have enjoyed much success and broad adoption in the commercial sector. But business and mission goals are larger than a single development team, and applying agile at scale is challenging along several dimensions. These recommended practices, orchestrated together, will help enable agility at scale:

1. Use Scrum of Scrums carefully when coordinating multiple teams.
2. Use an architectural runway to manage technical complexity.
3. Align feature-based development and system decomposition.
4. Use quality-attribute scenarios to clarify architecturally significant requirements.
5. Use test-driven development for early and continuous focus on verification.
6. Use end-to-end testing for early insight into emerging system properties.
7. Use continuous integration for consistent attention to integration issues.
8. Consider technical debt management as an approach to strategically manage system development.
9. Use prototyping to rapidly evaluate and resolve significant technical risks.
10. Use architectural evaluations to ensure that architecturally significant requirements are addressed.

Challenges to implementing the practices include large team sizes, high system complexity (e.g., due to stringent real-time, high-availability, and security requirements), and long development and operation cycles, which may require more attention to redesign and documentation than a strict reading of the agile principles might suggest. Enablers include a technical infrastructure that enables teams to collaborate, a management culture that empowers and trusts team decisions, and ensuring that all project activities are visible to team members and stakeholders. The reference section provides more resources on nine subtopics related to using agile at scale.

[2] Recommended practices: https://www.thecsiac.com/spruce/resources/ref_documents/agile-scale-aas-spruce-sei; webinar: http://vimeo.com/81636494.

These recommended practices—as well as challenges, enablers, and resources—for agile at scale, are available in full on the CSIAC SPRUCE website.[2] In December 2013, Rod Nord also presented a webinar on this topic to supplement the recommended practices and provide additional information about technical debt management.

Safety-Critical Systems

The recommended practices for *Safety-Critical Systems* were published in October 2013.[3] The practices were based on interviews with Peter Feiler, Julien Delange, and Charles Weinstock. Dr. Feiler is the technical lead and author of the SAE AS-2C Architecture Analysis & Design Language (AADL) standard and received the Carnegie Science Award for Information Technology for this work. Dr. Delange led research at the European Space Agency on software and system architectures. And Dr. Weinstock has conducted and published research in model-based verification, fault-tolerant computing, distributed real-time systems, and assurance cases.

For safety-critical systems, failure may cause serious injury to people, damage to equipment, or environmental harm. As the needs for real-time and fail-safe performance become more pervasive and stringent, it becomes harder to develop and evolve such systems. These recommended practices help an organization successfully develop and sustain safety-critical systems. The practices, orchestrated together, will help enable safety-critical systems:

1. Use quality attribute scenarios and mission-thread analyses to identify safety-critical requirements.
2. Specify safety-critical requirements, and prioritize them.
3. Conduct hazard and static analyses to guide architectural and design decisions.
4. Specify the architecture incrementally, in a formal notation.
5. Sustain a virtual integration of the software through multiple levels of its specification.
6. Use AADL to formally specify requirements and architecture.
7. Monitor implementation, integration, and testing.
8. Prepare a safety case for certification concurrent with developing the system.

Challenges to implementing the practices include the increasing scope of today's safety-critical systems; the number of interfaces among such systems and the environment; and the need for real-time, fail-safe performance. Enablers include nurturing a culture of safety across the organization; a discipline for architecture-centric engineering; a technical infrastructure that supports formal specifications and automated analyses; and engagement with model-based engineering communities that are learning how to conduct virtual integration and develop assurance cases. The reference section provides more resources on six subtopics related to safety-critical systems.

[3] Recommended practices: https://www.thecsiac.com/spruce/resources/ref_documents/safety-critical-sc-systems-spruce-sei; webinar: http://vimeo.com/86733718.

These recommended practices—as well as challenges, enablers, and resources—for developing safety-critical systems, are available in full on the CSIAC SPRUCE website.[3] In February 2014, Peter Feiler also presented a webinar on this topic to supplement the recommended practices and provide additional information about requirements specification, safety and reliability analysis, and virtual integration.

Monitoring Software-Intensive System Acquisition Programs

The recommended practices for *Monitoring Software-Intensive System Acquisition Programs* were published in November 2013.[4] The practices were based on interviews with Robert Ferguson. Mr. Ferguson is a Project Management Professional with 30 years of software-development experience in five different industries, specializing in pre-Milestone A cost estimation, and conducts research in software sustainment.

Effective program management requires maintaining an accurate understanding of a program's status, quickly identifying issues that threaten program objectives, and dealing with them efficiently. These recommended practices implement a "program dashboard" that helps the program manager and contractor come to a mutual understanding of a program's progress and the significance of deviations from expectations. The practices, orchestrated together, enable monitoring of software-intensive system acquisition programs:

1. Address management measures and their use in Requests for Proposals and contracts.
2. Set up the dashboard.
3. Assign and train staff to interpret the dashboards.
4. Populate and update the dashboard regularly and when needed.
5. Include discussion of dashboard changes in program reviews and as needed.
6. Refresh measures for each new phase/milestone.
7. Set new expectations and negotiate changes to commitments with stakeholders.

Challenges to implementing the practices include contractors who lack understanding of a program manager's commitments to the program's diverse stakeholders, program managers who lack understanding of contractors' data, and the difficulty for both in changing to new sets of measures. Enablers include selecting capable contractors, staffing a capable program management team, and implementing an infrastructure for measurement data on both sides. The reference section provides more resources on monitoring software-intensive system acquisition programs.

These recommended practices—as well as challenges, enablers, and resources—for monitoring software-intensive system acquisition programs, are available in full on the CSIAC SPRUCE website.[4]

[4] https://www.thecsiac.com/spruce/resources/ref_documents/monitoring-software-intensive-system-acquisition-sisa-engineering-programs-spr

Managing Intellectual Property in the Acquisition of Software-Intensive Systems

The web page for *Managing Intellectual Property in the Acquisition of Software-Intensive Systems* was published in December 2013.[5] The page was based on interviews with a senior member of the technical staff who chose to be anonymous but who has 20 years of experience supporting the software-acquisition practices of government programs.

Department of Defense regulations now require that programs develop an intellectual property (IP) strategy as part of the acquisition strategy [DoD 2013]. Who owns particular IP rights can facilitate, hinder, or even obstruct necessary changes to the software by the government. The recommended practices focus on managing rights in IP for acquisitions, with emphasis on noncommercial software. The practices, orchestrated together, will help enable managing intellectual property in the acquisition of software-intensive systems:

1. Determine the program's needs for IP rights throughout the acquisition life cycle.
2. Establish and maintain an IP strategy as part of the broader acquisition strategy.
3. Determine what kind of rights you need.
4. Address rights in IP in the source-selection process.
5. Address rights in IP and deliverables in contract negotiations and in the contract.
6. Protect rights in IP assets.
7. Conduct IP audits.

Challenges to implementing the practices include program managers' lack of understanding legal terminology, stakeholders' lack of understanding the context for acquisition, and the difficulty of determining appropriate data rights. Enablers include establishing priorities for the management of intellectual property and developing relationships with those who have expertise in intellectual property. The reference section provides more resources on five topics related to managing intellectual property in the acquisition of software-intensive systems.

These recommended practices—as well as challenges, enablers, and resources—for managing intellectual property in the acquisition of software-intensive systems, are available in full on the CSIAC SPRUCE website.[5]

Managing Operational Resilience

The web page for *Managing Operational Resilience* was published in March 2014.[6] The page was based on interviews with Julia H. Allen, Pamela Curtis, and Nader Mehravari. Ms. Allen has more than 20 years of experience conducting research in operational resilience and security frameworks and is the

[5] https://www.thecsiac.com/spruce/resources/ref_documents/recommended-practices-managing-intellectual-property-acquisition-software-in

[6] https://www.thecsiac.com/spruce/resources/ref_documents/recommended-practices-managing-operational-resilience

author and coauthor of several books on software security. Ms. Curtis has more than 25 years of experience in the information technology domain and conducts research and develops models and assessments related to improving and measuring operational resilience. Dr. Mehravari has more than 30 years of experience in the aerospace, defense, and telecommunications industries, including research in operational resilience, cybersecurity, and protection of critical infrastructure.

Organizations have invested a tremendous amount of resources in cybersecurity, yet cyber attackers continue to penetrate systems. Therefore, an organization should pursue a strategic approach that goes beyond activities that protect assets (e.g., by preventing attacks) to also include activities that sustain services and operations after an attack. Managing operational resilience includes all the practices of planning, integrating, executing, and governing these protection and sustainment activities. These recommended practices, orchestrated together, will help enable managing operational resilience:

1. Oversee and manage the execution of resilience activities throughout the organization.
2. Train staff at all levels in how to perform their assigned roles when disruptions occur.
3. Establish and maintain communications with stakeholders.
4. Identify, analyze, and mitigate risks to assets that could affect the delivery of high-value services.
5. Plan end-to-end handling of disruptive events from detection, to triage, to resolution.
6. Ensure the continuity of essential operations and services during and following disruptive events.
7. Identify, protect, and maintain critical assets.
8. Identify and manage dependencies on external entities, such as the supply chain.
9. Ensure that software that enables or performs the delivery of critical services and operations satisfies resilience requirements.

Challenges to implementing the practices include organizations' failure to understand that operational resilience requires a long-term commitment; organizations' failure to understand the full context of implementing operational resilience, which requires planning, coordination, and training across many interdependent domains; and overcoming organization hurdles to managing operational resilience. Enablers include coordinating the implementation of the practices across the organization (e.g., are everyone's roles in disaster recovery understood?), maintaining currency with relevant standards, and understanding compliance issues. The reference section provides more resources on seven topics related to managing operational resilience.

These recommended practices—as well as challenges, enablers, and resources—for managing operational resilience, are available in full on the CSIAC SPRUCE website.[6]

Evaluating Impact

We made an early decision to collect sufficient web data that would enable us to assess the utility of the web pages at a high level. We tracked the number of page views as a measure of visits to the website, the number of page views over time to help us assess what increased traffic to the website, and the amount of time visitors spent on each page as a measure of engagement with the material. The website also uses a Comments feature, which we hoped would provide us with more feedback about the topics' structure and content.

Table 1 summarizes total page views as of June 3, 2014, for each set of recommended practices. Little promotion occurred, yet the first three web pages each garnered more than 5,000 views and the last web page achieved almost 3,000 views in less than half the time as the first pages.

Table 1: Summary of Page Views for the Recommended Practices Web Pages

Recommended Practices	Publish Date	Page Views[a]
Agile at Scale	Sep. 2013	5,588
Safety-Critical Systems	Oct. 2013	5,342
Monitoring Software-Intensive System Acquisition Programs	Nov. 2013	5,519
Managing Intellectual Property in the Acquisition of Software-Intensive Systems	Dec. 2013	4,894
Managing Operational Resilience	Mar. 2014	2,947

[a] Page views are from data provided by Quanterion's content management system.

We tracked the total number of page views for the recommended practices web pages over time, shown in Figure 1. The metrics provide some insight into what activities drew traffic to the website. Traffic reached its first noticeable peak on November 11, 2013, after Quanterion sent an email to registered CSIAC members promoting the *Agile at Scale* web page. Traffic reached its highest peak on January 22, 2014, after Quanterion sent an email to registered CSIAC members promoting the practices and giving prominence to the *Safety-Critical Systems* page. CSIAC hosted a webinar related to *Agile at Scale* on December 11, 2013, and a webinar related to *Safety-Critical Systems* on February 12, 2014. The weeks after both webinars also show increases in visitor traffic to the recommended practices pages.

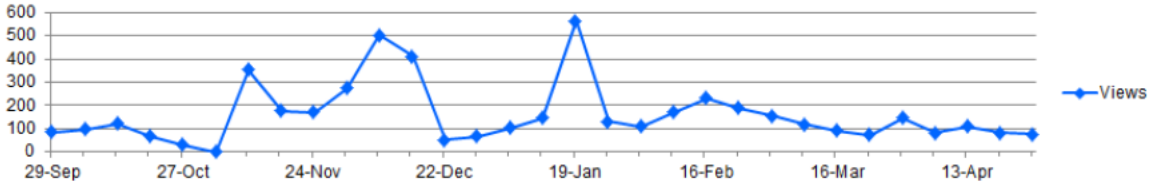


Figure 1: Total Page Views from Google Analytics for All Recommended Practices Web Pages

We tracked the amount of time visitors spent on each page as a measure of engagement. Time on page varied from two minutes for *Managing Intellectual Property in the Acquisition of Software-Intensive Systems* to about four minutes for *Monitoring Software-Intensive System Acquisition Programs*. Of course, the latter page is also longer, but four minutes represents strong user engagement with the material.

Each web page has a Comments feature at the bottom, but visitors did not use it. A moderator opened a discussion about one set of practices on the website, but two commenters produced only three comments

[CSIAC 2013]. The lack of comments may in part be due to the length of the web pages, placing the Comments feature well below the scroll, out of sight and mind for most readers. When Quanterion posted the much shorter Challenge Problem and Candidate Solution web pages related to the *Agile at Scale* web page, 12 comments resulted. Making the commenting feature more obvious, such as near the top of the web pages, was an aspect of web-page design that we had hoped to resolve prior to promoting the website. (For more detail, see Lesson 5 in the next section.)

Lessons Learned About Curating Best Practices

Our approach to the SPRUCE effort involved building a community around the broader SPRUCE website by synthesizing recommended practices on topics important to software acquisition and development. Our initial focus, consistent in many ways with earlier phases of the SPRUCE project, was to provide new but related content and to use promotion and other mechanisms for engaging the broader community to discuss that content, which would then build the volume of engagement. We highlight five lessons for curating recommended practices that we have drawn from our experience and that may benefit others involved in similar pursuits.

Lesson 1. Develop relationships with domain experts to increase the quality of the output.

Creating a web page that distilled a software topic into 10 or fewer recommended practices required our authors not only to be experts in their fields but also to prepare to work with us. When contacting interviewees, we provided guidance on how we could get the most out of our one-hour interview sessions. We asked them to think about why the topic is challenging, what are the top practices, what enables an organization to achieve effective results with those practices, and the best materials for further reading. Usually, this worked well. For one topic, we did not sufficiently narrow the scope of the interview and thus found it difficult to distill the practices. We then needed to interview another expert to complete the piece. After we wrote the practices from interview notes and supplemental material, we asked interviewees to review and revise the practices. Again, this usually worked well. For one topic, two experts whom we did not initially interview, but in hindsight should have, entered the process late in the review phase, and we re-scoped the practices as a result of their input. But for each topic, we eventually came to agreement on the scope of the recommended practices. In all cases, developing relationships with our interviewees facilitated the content-development process, including when we needed more help from the experts later to address comments or incorporate additional information.

Lesson 2. Use a structured interview method to gather more information in less time.

Before each interview, we researched the topic and rarely found a clear, prescriptive set of practices focused only on that topic. By recruiting the right technical experts from within the SEI, we found that when given time to distill their experiences into a set of practices, and when asked to talk through them, they could often produce such a set. The interview method relieves interviewees of the burden to structure the information for presentation and encourages them to consider the topic from multiple perspectives. Later, we compared what the experts said against multiple non-SEI sources and made some adjustments to the recommended practices (and challenges, enablers, and resources), but the interview—together with the prepared interviewee—provided a valuable mechanism to quickly elicit most of the points to be made in the recommended practices for that topic.

Lesson 3. Develop trust with external collaborators to enable progress in spite of different goals.

In contributing to the SPRUCE website, the SEI worked within the constraints of an existing website design. SPRUCE had developed around the central idea of Challenge Problems and Candidate Solutions, and the website was designed to facilitate this problem–solution model. The SEI's recommended practices introduced a new component into this model. We worked with Quanterion to design a user-friendly space on the SPRUCE website for the practices, and we maintained transparency about our differing project goals. Over the fall and winter of FY14, Quanterion worked with the SEI to accommodate the recommended practices, to make the practice topics easier for visitors to find, and to make the website easier to navigate. Some issues with navigability remain, and we compromised on website design, given the host's existing web-page templates. However, developing a good working relationship with our collaborators enabled us to make progress on the SPRUCE project in spite of our starting the project with different goals for using the SPRUCE website.

Lesson 4. Align web metrics tools across stakeholders and agree on what to measure.

The CSIAC website uses a content management system (CMS) that employs a method of counting page visits that varies significantly from that used by Google Analytics. The CMS counts caching as a visit, and it does not filter out visits from bots. Thus, it records more events than we'd like to report as visits. In addition, we understandably did not have access to the CMS, so we could not drill down into the data to investigate the details; instead, CSIAC provided screenshots of the aggregate data. Google Analytics, on the other hand, undercounts views and visits. If a user visits a web page but does not browse to a second web page within the same website, Google Analytics counts this visit as a bounce, regardless of the length of the visit [Google 2014]. For most of the SEI's performance period, from any one of the practices web pages, there was no indication for visitors that other practices pages existed, a navigation issue that may have contributed to large bounce rates, even if visitors read the entire web page before leaving the site. We determined to use primarily Google Analytics for our reporting, given our greater experience with what it provides, but referring to the CSIAC's metrics gave us another perspective on activity not counted by Google Analytics, some of which we wanted to count and some not. The problem of reconciling two data sources was compounded by the loss of our web analytics expert mid-way through the performance period. Ideally, since we could not use the same tool that the CSIAC used, we would have worked out a way to receive more data from the site host's CMS and located another source of analytics expertise.

Lesson 5. When the vision or expected use for a website changes, extensive website redesign may be necessary.

The recommended practices were to be hosted on a website originally designed to meet a different purpose that was defined in an earlier phase of SPRUCE project. This constrained redesigning the website to make it easier for site visitors to find the recommended practices and navigate among them. The Challenge Problems and Candidate Solutions sections of the website, resulting from this earlier phase of the SPRUCE project, were easy to find from the main navigation menu, while the menu listing the recommended practices resulting from the new phase of the SPRUCE project appeared in the right-hand column after a short scroll down the page. And as mentioned in Lesson 4, from any one of the practices web pages, there was no indication to visitors that other practices pages existed. We decided to delay promoting the recommended practices until after the SPRUCE website could better accommodate visitors to the practices. From November 2013 to March 2014, the organization responsible for hosting the website incrementally made design changes so that the recommended practices on the SPRUCE landing page and navigation among all the practices web pages were more visible. At the end of February, the website redesign was sufficiently complete to welcome and inform visitors. Although we still wanted to improve access to comments and discussion features, we began developing a promotion campaign.

CONCLUSION

Current Status

The SEI's role in the larger SPRUCE effort was to curate recommended practices on topics related to software engineering and to explore and assess an alternative approach to engage a community. We developed a process for identifying and curating recommended practices and applied that process to create web pages on five important software topics: *Agile at Scale*, *Safety-Critical Systems*, *Monitoring Software-Intensive System Acquisition Programs*, *Managing Intellectual Property in the Acquisition of Software Intensive Systems*, and *Managing Operational Resilience*. The CSIAC also hosted two webinars presented by our collaborating experts from the SEI.

We tracked several web metrics—including page views, views over time, and average time on page—to help substantiate the relevance of the practices and gauge community interest. Quanterion addressed many of our suggestions for increasing the potential impact of the recommended practices by making the SPRUCE website easier to navigate and by making both the series and individual practices easier for visitors to find. Our web metrics provide evidence of community interest in the identification of recommended practices that can address the challenges encountered in software-intensive system development and acquisition. We also began planning a promotion campaign for the practices series to increase traffic to the SPRUCE website. Unfortunately, funding for the SEI SPRUCE project ended before we could begin implementing it.

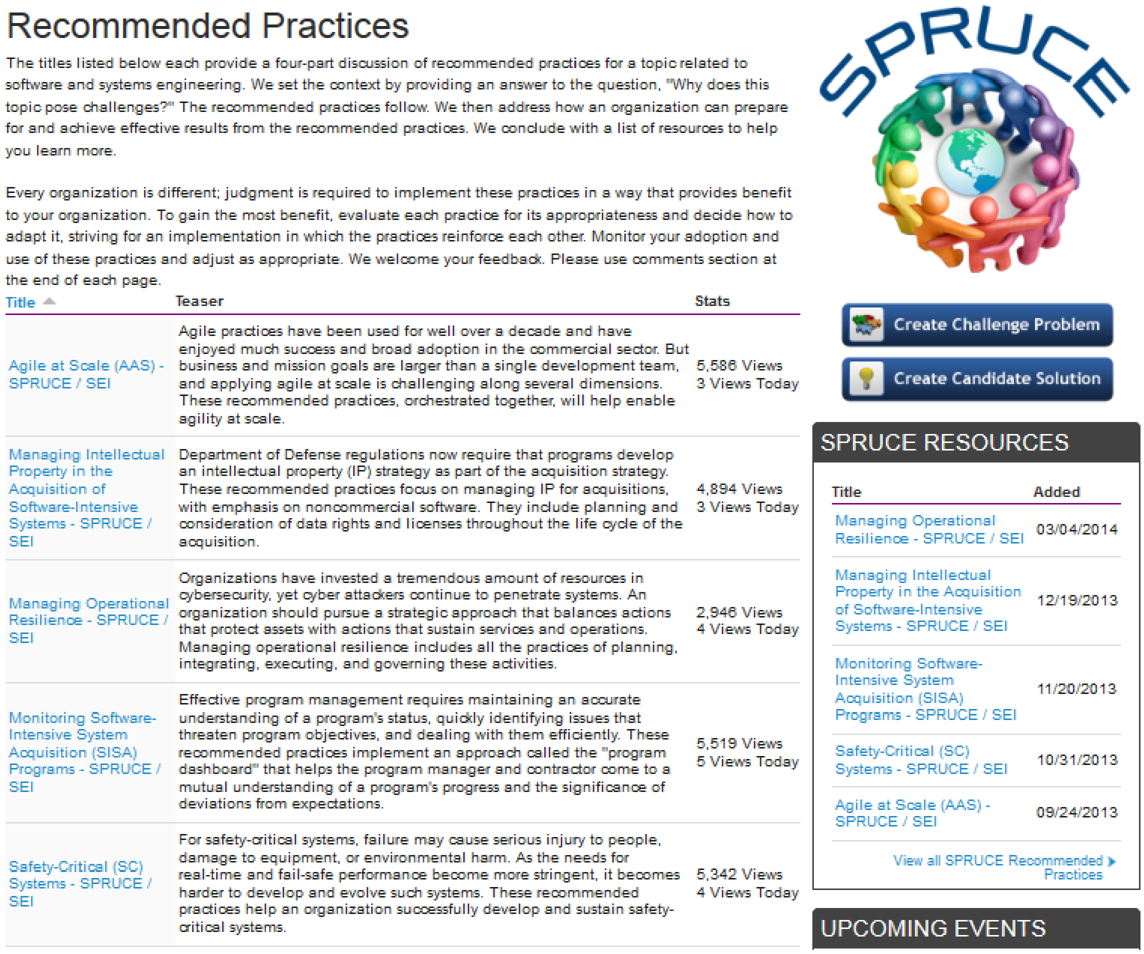
Proposed Next Steps

Should we have further opportunity to contribute to SPRUCE, we have identified proposed next steps to continue the SEI's leadership role in transitioning recommended practices for software-intensive systems. In the short term, the SEI should consider hosting the recommended practices on the SEI website. This is within the SEI's mission of transitioning maturing solutions to widespread use. In assessing future transition mechanisms, the SEI should address the questions "To what extent should we engage a broad community in identifying, discussing, questioning, and developing recommended practices?" and "What role should ideation and discussion forums play in SEI's technology transition efforts more generally?" Technology transition is a key part of the SEI mission. Engaging a broader community in the curation of recommended practices and technologies may improve the quality of the practices and hasten their use by the software engineering community broadly.

In the long term, there are several possible ways to extend this work. First, there are many more software topics for which we could work with technical experts to gather recommended practices. Second, an ongoing promotion campaign could help make a broader range of government, university, and defense-industry communities aware of such continuously growing content and could help encourage them to use it, add to it, and help revise it. Tracking web metrics in greater detail will also be useful in measuring community engagement. Developing recommended practices, with community engagement, is a means to broadly obtain acceptance that could improve the quality of DoD software-intensive systems. Such an approach could complement the Institute of Electrical and Electronics Engineers' standards development by drawing input from a broad community of expertise and providing early indication of when a topic is ready for de jure standardization.

APPENDIX THE SPRUCE/SEI WEBSITE

The following screenshots capture some views of the SPRUCE/SEI website. Figure 2 shows the landing page for the recommended practices, Figure 3 shows a sample page for one of the topics, and Figure 4 shows a table of contents with the standard four-part structure that we used for each of the recommended practices.


SPRUCE
Create Challenge Problem
Create Candidate Solution
View all SPRUCE Recommended Practices

Recommended Practices

The titles listed below each provide a four-part discussion of recommended practices for a topic related to software and systems engineering. We set the context by providing an answer to the question, "Why does this topic pose challenges?" The recommended practices follow. We then address how an organization can prepare for and achieve effective results from the recommended practices. We conclude with a list of resources to help you learn more.

Every organization is different; judgment is required to implement these practices in a way that provides benefit to your organization. To gain the most benefit, evaluate each practice for its appropriateness and decide how to adapt it, striving for an implementation in which the practices reinforce each other. Monitor your adoption and use of these practices and adjust as appropriate. We welcome your feedback. Please use comments section at the end of each page.

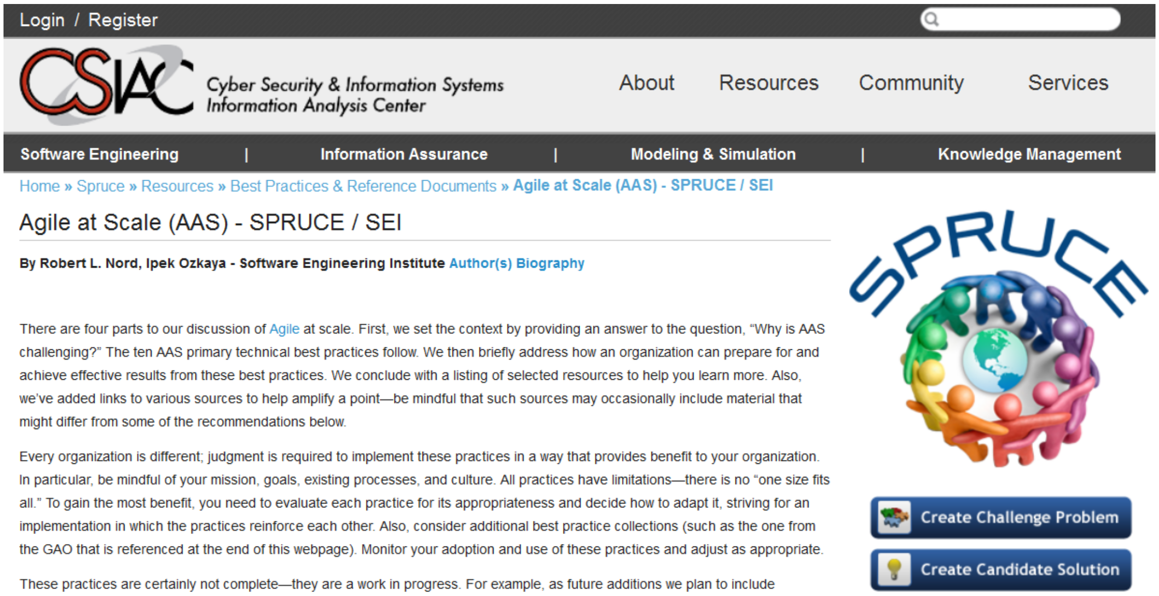
Title ▲	Teaser	Stats
Agile at Scale (AAS) - SPRUCE / SEI	Agile practices have been used for well over a decade and have enjoyed much success and broad adoption in the commercial sector. But business and mission goals are larger than a single development team, and applying agile at scale is challenging along several dimensions. These recommended practices, orchestrated together, will help enable agility at scale.	5,586 Views 3 Views Today
Managing Intellectual Property in the Acquisition of Software-Intensive Systems - SPRUCE / SEI	Department of Defense regulations now require that programs develop an intellectual property (IP) strategy as part of the acquisition strategy. These recommended practices focus on managing IP for acquisitions, with emphasis on noncommercial software. They include planning and consideration of data rights and licenses throughout the life cycle of the acquisition.	4,894 Views 3 Views Today
Managing Operational Resilience - SPRUCE / SEI	Organizations have invested a tremendous amount of resources in cybersecurity, yet cyber attackers continue to penetrate systems. An organization should pursue a strategic approach that balances actions that protect assets with actions that sustain services and operations. Managing operational resilience includes all the practices of planning, integrating, executing, and governing these activities.	2,946 Views 4 Views Today
Monitoring Software-Intensive System Acquisition (SISA) Programs - SPRUCE / SEI	Effective program management requires maintaining an accurate understanding of a program's status, quickly identifying issues that threaten program objectives, and dealing with them efficiently. These recommended practices implement an approach called the "program dashboard" that helps the program manager and contractor come to a mutual understanding of a program's progress and the significance of deviations from expectations.	5,519 Views 5 Views Today
Safety-Critical (SC) Systems - SPRUCE / SEI	For safety-critical systems, failure may cause serious injury to people, damage to equipment, or environmental harm. As the needs for real-time and fail-safe performance become more stringent, it becomes harder to develop and evolve such systems. These recommended practices help an organization successfully develop and sustain safety-critical systems.	5,342 Views 4 Views Today

SPRUCE RESOURCES

Title	Added
Managing Operational Resilience - SPRUCE / SEI	03/04/2014
Managing Intellectual Property in the Acquisition of Software-Intensive Systems - SPRUCE / SEI	12/19/2013
Monitoring Software-Intensive System Acquisition (SISA) Programs - SPRUCE / SEI	11/20/2013
Safety-Critical (SC) Systems - SPRUCE / SEI	10/31/2013
Agile at Scale (AAS) - SPRUCE / SEI	09/24/2013

UPCOMING EVENTS

Figure 2: The Index Page for the Recommended Practices


Login / Register
Cyber Security & Information Systems Information Analysis Center
About
Resources
Community
Services
Software Engineering | Information Assurance | Modeling & Simulation | Knowledge Management
Home » Spruce » Resources » Best Practices & Reference Documents » Agile at Scale (AAS) - SPRUCE / SEI
SPRUCE
Create Challenge Problem
Create Candidate Solution

Agile at Scale (AAS) - SPRUCE / SEI

By Robert L. Nord, Ipek Ozkaya - Software Engineering Institute Author(s) Biography

There are four parts to our discussion of Agile at scale. First, we set the context by providing an answer to the question, "Why is AAS challenging?" The ten AAS primary technical best practices follow. We then briefly address how an organization can prepare for and achieve effective results from these best practices. We conclude with a listing of selected resources to help you learn more. Also, we've added links to various sources to help amplify a point—be mindful that such sources may occasionally include material that might differ from some of the recommendations below.

Every organization is different; judgment is required to implement these practices in a way that provides benefit to your organization. In particular, be mindful of your mission, goals, existing processes, and culture. All practices have limitations—there is no "one size fits all." To gain the most benefit, you need to evaluate each practice for its appropriateness and decide how to adapt it, striving for an implementation in which the practices reinforce each other. Also, consider additional best practice collections (such as the one from the GAO that is referenced at the end of this webpage). Monitor your adoption and use of these practices and adjust as appropriate.

These practices are certainly not complete—they are a work in progress. For example, as future additions we plan to include

Figure 3: Sample Web Page for the Recommended Practices

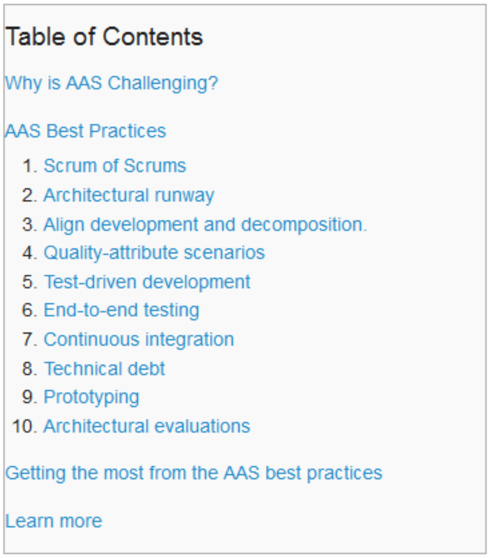


Table of Contents

Why is AAS Challenging?

AAS Best Practices

1. Scrum of Scrums
2. Architectural runway
3. Align development and decomposition.
4. Quality-attribute scenarios
5. Test-driven development
6. End-to-end testing
7. Continuous integration
8. Technical debt
9. Prototyping
10. Architectural evaluations

Getting the most from the AAS best practices

Learn more

Figure 4: Sample Contents for the Recommended Practices

REFERENCES

[CSIAC 2013]

Cyber Security and Information Systems Information Analysis Center. *What Has Been Your Experience on the Scalability of Agile?* CSIAC, 2013. https://www.thecsiac.com/discussion/what-has-been-your-experience-scalability-agile

[CSIAC 2014]

Cyber Security and Information Systems Information Analysis Center. *Systems and Software Producibility Collaboration Environment (SPRUCE)*. CSIAC, 2014. https://www.thecsiac.com/spruce

[DoD 2013]

Department of Defense. Interim DoD Instruction 5000.02, *Operation of the Defense Acquisition System*. November 25, 2013.

[Google 2014]

Google Analytics. *Exit Rate vs. Bounce Rate*. Google, 2014.

[Lardieri 2009]

Lardieri, P.; Buskens, R.; Srinivasan, S.; McKeever, W.; & Drager, S. “SPRUCE: A Web Portal for the Collaborative Engineering of Software Intensive Systems Producibility Challenge Problems and Solutions,” 276–283. *Proceedings of the International Symposium on Collaborative Technologies and Systems*. Baltimore, MD, May 2009. IEEE Computer Society Press, 2009.

[Lockheed Martin 2013]

Lockheed Martin ATL, Booz Allen Hamilton, Vanderbilt University, & Lockheed Martin Aeronautics. *Software and Systems Producibility Collaboration and Experimentation Environment (SPRUCE)*. Lockheed Martin Advanced Technology Laboratories, 2013.

[SPRUCE 2014]

Systems and Software Producibility Collaboration Environment (SPRUCE). *Recommended Practices*. CSIAC, 2014. https://www.thecsiac.com/spruce/recommended_practices

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM-0001764